

WEB SERVICES

With Axios & Vue



Can You?

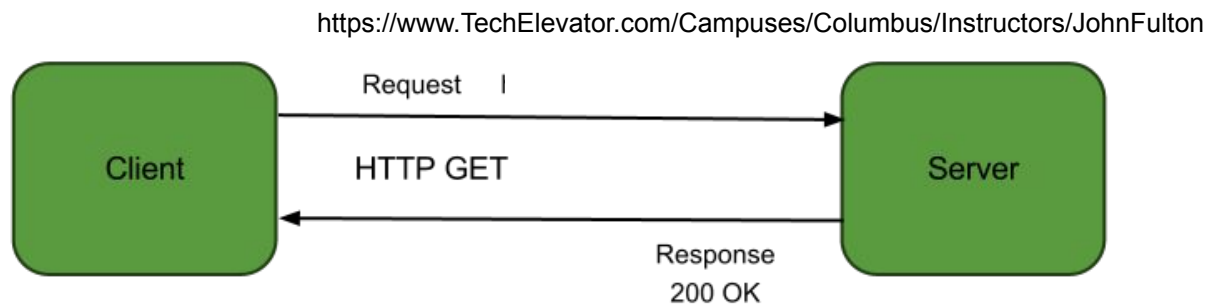
- Explain the process of a typical HTTP request between a web browser and a server
- Explain what a GET request is used for
- Explain that a 2xx Status Code indicates "success"
- Make an HTTP GET request using Postman and inspect the result
- Explain what JSON is and use it in a JavaScript program
- Make an HTTP GET request to a RESTful web service using the Axios library and process the response
- Build a service object for interacting with a RESTful web service
- Use the Vue lifecycle hook `created()` to call a web service to retrieve data when a view is rendered
- Explain the difference between synchronous and asynchronous code
- Explain what a promise is and how it works
- Explain why asynchronous coding techniques are frequently used in JavaScript for interacting with server-side components

REST

REST

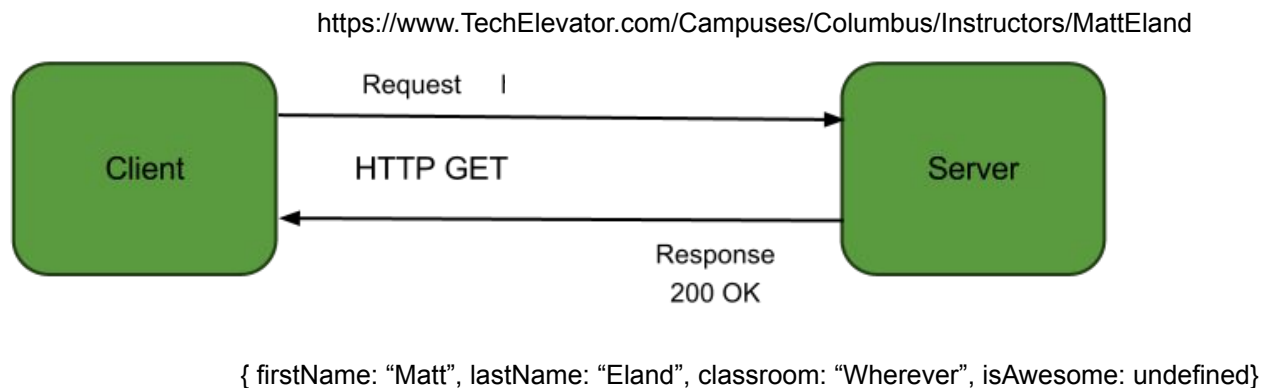


REST

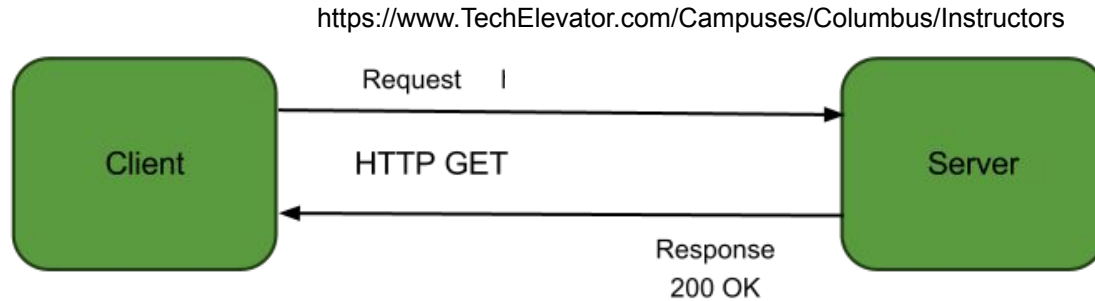


`{ firstName: "John", lastName: "Fulton", classroom: ".NET", isAwesome: true }`

REST



REST



```
[  
  { firstName: "John", lastName: "Fulton", classroom: ".NET", isAwesome: true },  
  { firstName: "Matt", lastName: "Eland", classroom: "Wherever", isAwesome: undefined },  
  // ... more instructors here  
]
```

VERBS & RESOURCES

GET	/pet/{petId}	Find pet by ID
PUT	/pet	Update an existing pet
DELETE	/pet/{petId}	Deletes a pet
POST	/pet/{petId}/uploadImage	uploads an image

STATUS CODES



HTTP STATUS CODES

2xx Success

200 Success / OK

3xx Redirection

301 Permanent Redirect

302 Temporary Redirect

304 Not Modified

4xx Client Error

401 Unauthorized Error

403 Forbidden

404 Not Found

405 Method Not Allowed

5xx Server Error

501 Not Implemented

502 Bad Gateway

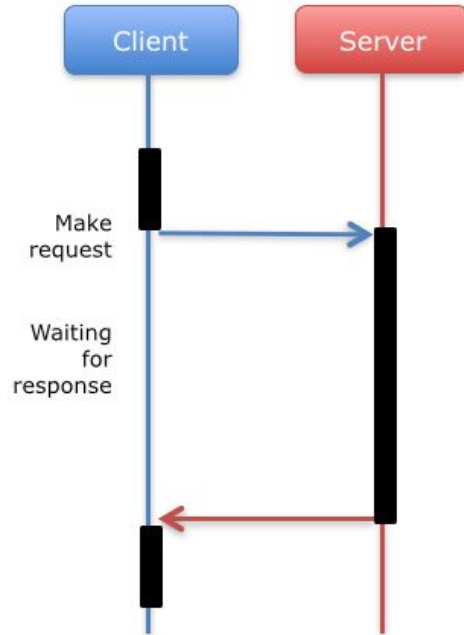
503 Service Unavailable

504 Gateway Timeout

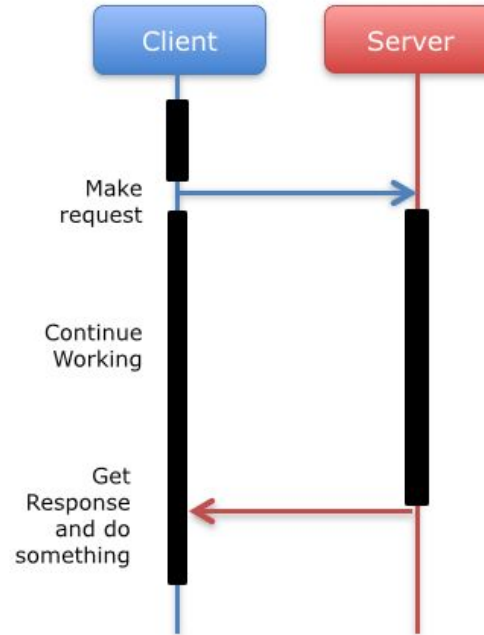
SYNCHRONOUS & ASYNCHRONOUS

SYNC VS ASYNC

Synchronous



Asynchronous

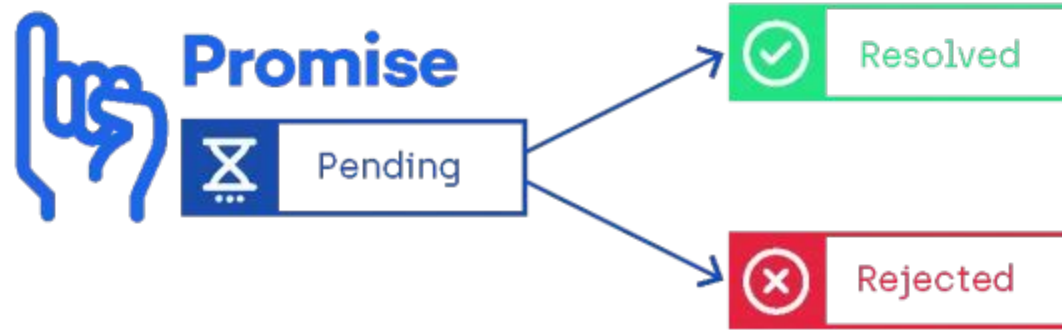


AXIOS

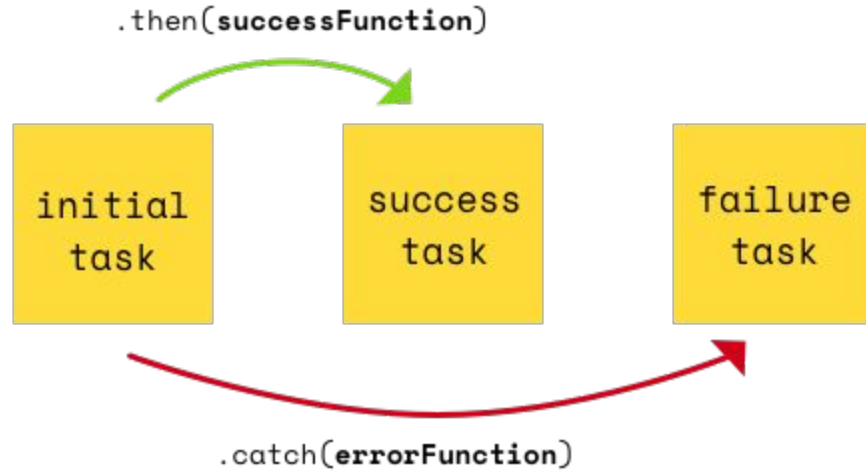
AXIOS GET

```
1 /**
2  * Gets all items on the server
3  * @returns {Promise} a promise that will complete with a list of items
4  */
5 getAllItems() {
6   // Create our Axios instance used to communicate with the server
7   const http = axios.create({
8     baseURL: 'https://some.website.com'
9   });
10
11   return http.get('/items'); // This is added to the end of baseURL specified above
12 }
```

WHAT IS A PROMISE?



USING A PROMISE



AXIOS GET

```
1 /**
2  * Gets all items on the server
3  * @returns {Promise} a promise that will complete with a list of items
4  */
5 getAllItems() {
6   // Create our Axios instance used to communicate with the server
7   const http = axios.create({
8     baseURL: 'https://some.website.com'
9   });
10
11   return http.get('/items'); // This is added to the end of baseURL specified above
12 }
```

```
1 getAllItems().then(response => {
2   // response.data is loaded from the contents of the response body
3   // It's typically going to be a JavaScript object or an array of objects
4   const items = response.data;
5   this.$store.commit('ITEMS_LOADED', items);
6 });
```

PROMISE ERROR HANDLING

```
1 getAllQuestions()  
2   .then(response => {  
3     // Data is loaded from the contents of the response body  
4     // It's typically going to be a JavaScript object or an array of objects  
5     const questions = response.data;  
6     this.$store.commit('QUESTIONS_LOADED', questions);  
7   })  
8   .catch(error => {  
9     console.error('An error occurred trying to load questions', error);  
10  })  
11  .finally(() => console.log('Finally!'));
```

SERVICES

```
1 import axios from 'axios';
2
3 // Create our Axios instance used to communicate with the server
4 const http = axios.create({
5   baseURL: 'https://some.url.net'
6 });
7
8 export default { // This object is what other files will import via the import keyword
9
10   getAllItems() {
11     return http.get('/items'); // This is added to the end of baseURL specified above
12   },
13
14   getItem(id) {
15     return http.get(`/items/${id}`);
16   },
17
18   update(myItem) {
19     return http.put(`/items/${myItem.id}`, myItem);
20   },
21
22   create(myItem) {
23     return http.post('/items', myItem);
24   },
25
26   delete(myItem) {
27     return http.delete(`/items/${myItem.id}`);
28   }
29
30 };
```

AXIOS VERBS

```
getItem(id) {  
  return http.get(`/items/${id}`);  
},  
  
update(myItem) {  
  return http.put(`/items/${myItem.id}`, myItem);  
},  
  
create(myItem) {  
  return http.post('/items', myItem);  
},  
  
delete(myItem) {  
  return http.delete(`/items/${myItem.id}`);  
}
```

**HAPPY
CODING!**

